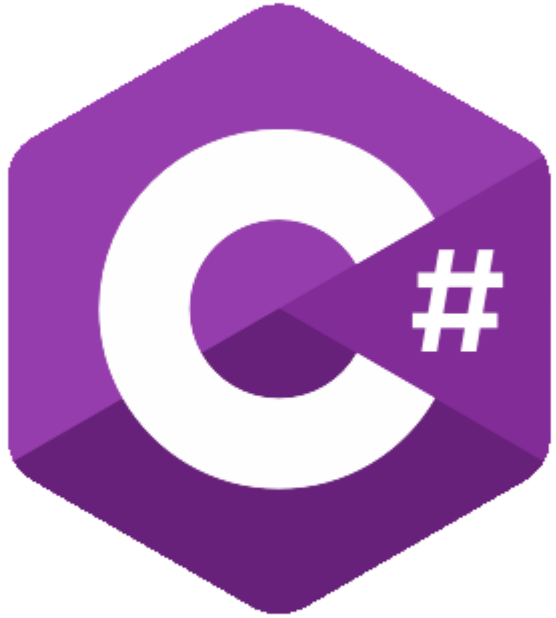
PV178 – Lab 03

Karel Jiránek


Kahoot!

Agenda

- Přetěžování metod
- Params
- Aliasy
- Nullovatelné typy
- Typovost a konverze typů
- Implicitní vs explicitní typování
- Dynamické typování
- Pole



Přetěžování metod

- Nelze podle návratového typu (proč?)
- Musí se lišit počet parametrů nebo jejich typ

Samostatná práce

- Úkol 1, 2 a 3 v projektu Params

Aliasy

- Zkracují zápis základních datových typů
- Používejte vždy když to jde!

Klíčové slovo v C#	*.NET type*
`bool`	`System.Boolean`
`byte`	`System.Byte`
`sbyte`	`System.SByte`
`char`	`System.Char`
`decimal`	`System.Decimal`
`int`	`System.Int32`
...	...

Nullovatelné typy

- Rozšiřují hodnotové typy - můžeme do nich přiřadit *null*
- Za datový typem je otazník

```
int? number = null;
```

- Note: dva otazníky za sebou znamenají „Null coalescing“ (česky propagade nullu - asi)

```
int? number1 = null;
int number2;

// Následující dva zápisy jsou významově stejné
number2 = number1 ?? 3;
number2 = number1 == null ? 3 : number1;
```

Typovost

- Pro získání typu objektu používáme metodu *GetType()*
- *typeof* nám vrací typ objektu, který je ze třídy tvořen

```
// Možnost 1 (old-school)
if (truckPosition.GetType() == typeof (Quarry))
{

}


// Možnost 2
if (truckPosition is Quarry))
{

}
```

Konverze typů

- Používáme kulaté závorky a typ objektu na který chceme přetypovat (možnost 1)
- Použijeme klíčové slovo *as* (možnost 2)
- Jaký je rozdíl? (viz příklad níže)

```
// Možnost 1 – vyhodí vyjímku, když nejde přetypovat
var quarry = (Quarry) truckPosition;


// Možnost 2 – vrátí null když nejde přetypovat
var quarry = truckPosition as Quarry;
```

Samostatná práce

- Úkol 1 a 2 v projektu TypesAndConversions

Implicitní vs explicitní typování

- Implicitní – používání datového typu
- Explicitní – používání klíčového slova *var* – reálný typ za nás doplní překladač
- Doporučení:
  - piště vždy *var*

```
int x = 10; //explicitne typovana premenna

var y = new List<object>(); //implicitne typovana premenna
```

Dynamické typování

- Typ proměnné je určen až za běhu programu
- Klíčové slovo *dynamic*
- Ztrácíte možnost InteliSense (nápověda od VS), typovou kontrolu
- Používejte jen když opravdu, ale opravdu musíte!
- JavaScript like code (eh...)

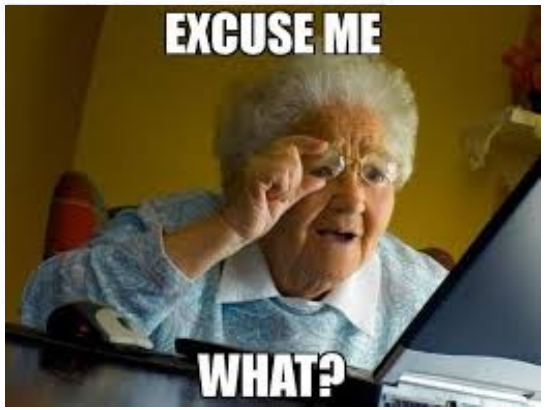
```
dynamic y = "hello world!";
```

Dynamický svět není tak úžasný! Nebo je?

- Když je svět dynamický a vše jde na všechno přetypovat


```
> typeof NaN
< "number"
> 9999999999999999
< 10000000000000000
> 0.5+0.1==0.6
< true
> 0.1+0.2==0.3
< false
> Math.max()
< -Infinity
> Math.min()
< Infinity
> []+[]
< ""
> []+{}
< "[object Object]"
> {}+[]
< 0
> true+true+true===3
< true
> true-true
< 0
```

```
> true==1
< true
> true===1
< false
> (!+[]+[]+![]).length
< 9
> 9+"1"
< "91"
> 91-"1"
< 90
> []==0
< true
```



Pattern matching

- Testování výrazu, jestli má určité charakteristiky

```
// 1
if (input is null)
{
        ...
}

// 2
abstract class Person{}
class Teacher: Person{ public int Salary { get; set; }}
class Student: Person{ public string Name { get; set; } }

switch (person)
{
        case Teacher teacher when teacher.Salary > 100_000: break; // Plný
zápis
        case Student { Name: "Josh" }: break; // Zkrácený zápis

        ...
}
```

Samostatná práce

- Úkol 1 až 4 v projektu Patterns

Pole

- Skupina proměnných **stejného** datového typu
- Přístup k jednotlivým položkám pomocí indexu (hranaté závorky)
- Číslováno od nuly
- Pevně daný rozměr
- Referenční datový typ
- Vytváří se přes klíčové slovo `new`
- Může být více rozměrné (.Rank vrací počet rozměrů pole)

Pole - metody

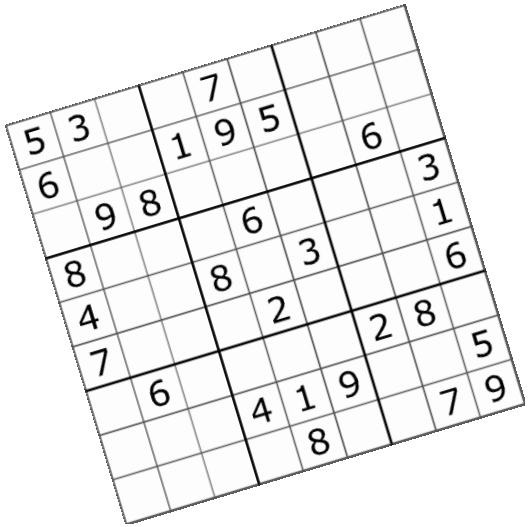
- kopírování polí – *CopyTo*
- řazení prvků v poli – *Sort*
- obrácení pořadí prvků – *Reverse*
- změna velikosti pole – *Resize*
- Nahrazení hraných závorek – *ElementAt* (rozšiřující metoda)
- Získání velikosti/délky jedné dimenze – GetLength(…)

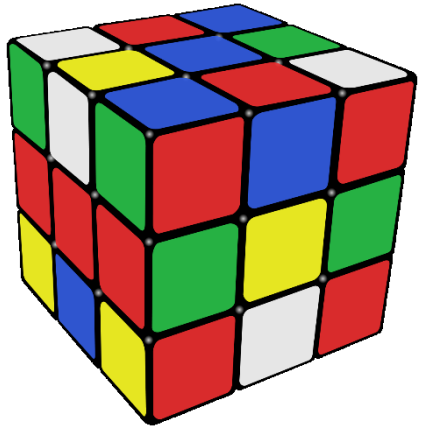
!!! Length vrací velikost celého pole - `new int[9, 9]` => Length je 81 !!!

Vícerozměrné pole

```
var matrix2 = new double[9, 9]; // dvojrozměrné
var matrix3 = new double[3, 3, 3]; // trojrozměrné

matrix2[0, 2] = 2;
matrix3[1, 2, 2] = 3;
```





Nepravidelné pole – Jagged Array

- Různá délku jednotlivých řádků
- Musí být pevně definován první rozměr pole

J	A	G	G	E	D			
A	R	R	A	Y	S			
A	R	E						
C	O	N	F	U	S	I	N	G

```
private int[][] crossword = new int[4][];
crossword[0] = new int [6];
crossword[1] = new int [6];
crossword[2] = new int [3];
crossword[3] = new int [9];

private int[,][] damagedRubikCube = new int[3, 3][];
damagedRubikCube[0, 0] = new int [2];
```

Samostatná práce

- Úkol 1 a 2 v projektu Arrays